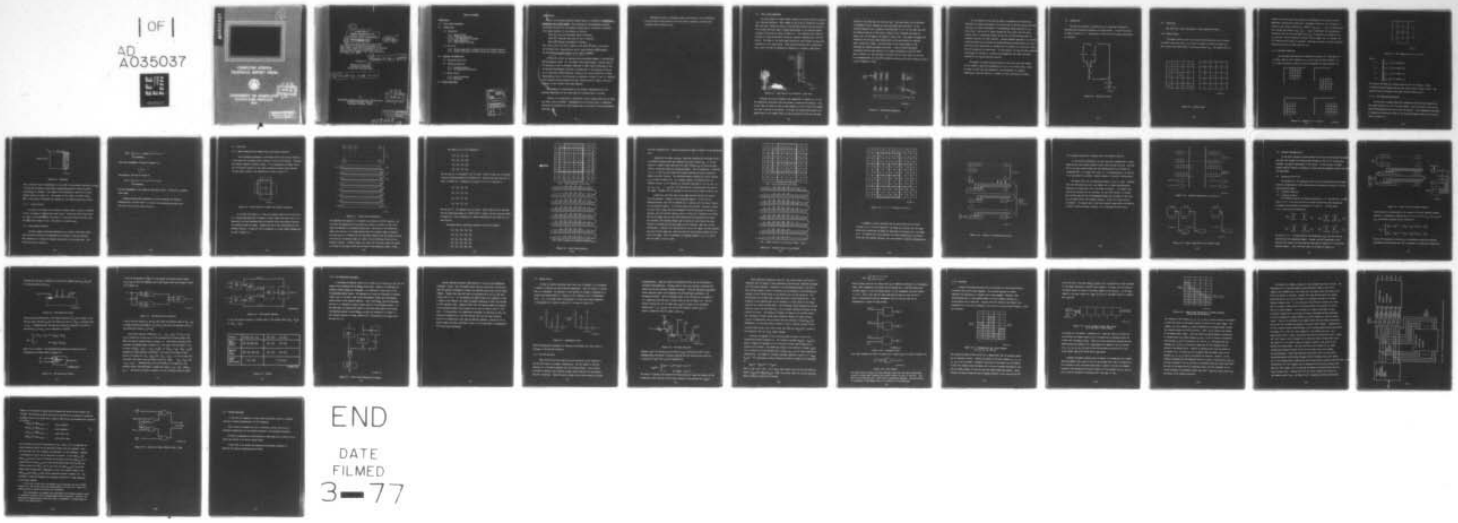
AD-A035 037 MARYLAND UNIV COLLEGE PARK COMPUTER SCIENCE CENTER F/G 17/5
A DISCUSSION OF HARDWARE IMPLEMENTATION FOR AN AUTOMATIC TARGET--ETC(U)
OCT 76 DAAG53-76-C-0138
UNCLASSIFIED NL

| OF |
AD
A035037



END

DATE
FILMED
3-77

B.S.

COMPUTER SCIENCE TECHNICAL REPORT SERIES

UNIVERSITY OF MARYLAND
COLLEGE PARK, MARYLAND
20742

(6)

A DISCUSSION OF HARDWARE IMPLEMENTATION FOR AN AUTOMATIC TARGET CUEING SYSTEM.

(11) 31 October 1976

(12) 43 p.

This is the second quarterly status report on a program for Recognition Technology for a Smart Sensor, conducted by Westinghouse for Maryland under Contract DAAG 53-76-C-0138 with the U. S. Army Electronics Command, Night Vision Laboratory, Ft. Belvoir, VA 22060.

(15) DAAG53-76-C-0138

Prepared for

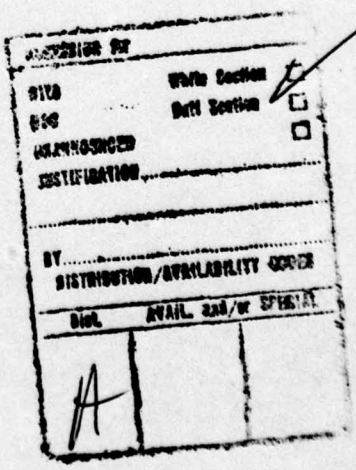
Computer Science Center
University of Maryland
College Park, Maryland 20742

(9) Quarterly rept. no. 2.

By

Westinghouse Defense and Electronic Systems Center
Systems Development Division
Baltimore, Maryland 21203

TABLE OF CONTENTS



INTRODUCTION

This is the second quarterly status report on a program for RECOGNITION TECHNOLOGY FOR A SMART SENSOR, being conducted by the Westinghouse Systems Development Division for the Computer Science Center, University of Maryland. The program consists of three phases, as follows:

Phase 1 - Task and Technology Review (3 months);

Phase 2 - Algorithm Selection and Test (9 months);

Phase 3 - Hardware Development (9 months).

This report covers the first 3 months of the Phase 2 effort. The report was prepared by Mr. Thomas Willett and Dr. Nathan Bluzer of Westinghouse. The Westinghouse program manager is Dr. Glenn E. Tisdale.

During the quarter six meetings were held between members of the Maryland and Westinghouse teams. Mr. John Dehne, NVL program manager, attended some of the meetings as well as holding a Symposium on Smart Sensor Technology at NVL. The Maryland-Westinghouse team made a presentation at that meeting as well as at the semi-annual DARPA Symposium, chaired by Maj. David Carlstrom, on Image Understanding, held at the University of Maryland on October 13 and 14. Westinghouse received further guidance from Maj. Carlstrom and Mr. Dehne in the performance of this contract from these Symposia.

Westinghouse is concentrating on the hardware implementation of the Maryland algorithms for the focal plane and treating them as a system.

Clearly, it is desirable to incorporate an auto cueing system into as small and cheap a box as possible. Implementation on the focal plane in compatible structures is desirable for these reasons and is the thrust of Second Generation FLIR work.

Westinghouse hopes to determine system partitioning by first attempting to put as much of the system on the focal plane as possible, starting with the lower level functions first.

1.0 FOCAL PLANE PROCESSING

The focal plane of a Smart Sensor contains the array of sensor elements, e.g., infrared detectors. Each element in the array is divided in two parts. The front part, facing the scene, is the infrared detector and the back part is an analog CCD which holds a charge proportional to the detected signal strength of the scene as shown in Figure 1-1. For FLIRS, the focal plane is located in a dewar flask which is kept at cryogenic temperatures of 20°-50° Kelvin. The leads to the focal plane electronics come through the dewar and therefore act as a heat source. Focal plane electronics are an additional heat source and must be designed for operation at cryogenic temperatures.

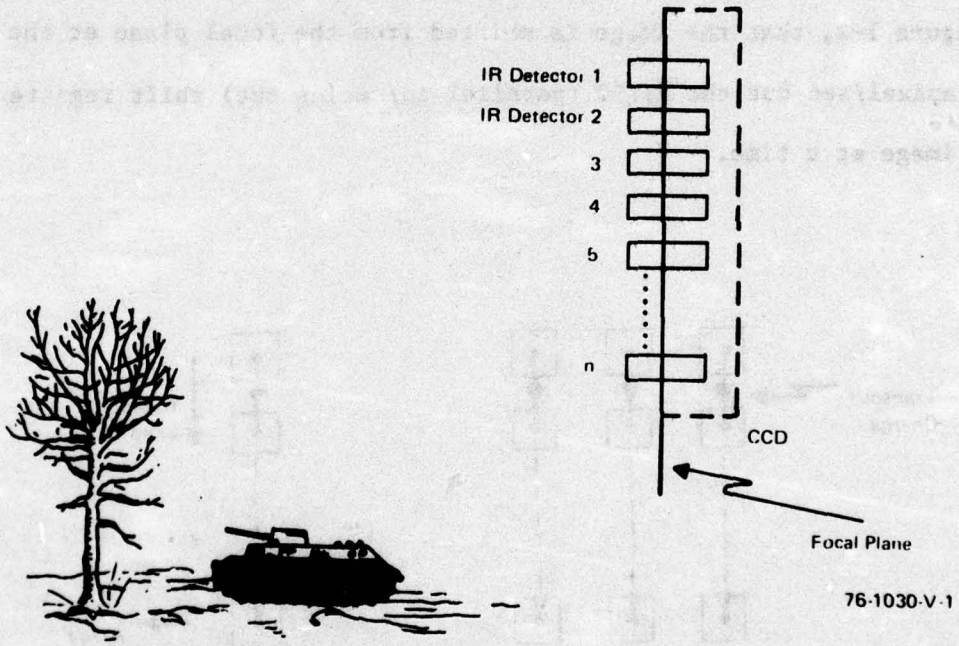


Figure 1-1. Side View of an IR Detector - CCD Array

Although the array is shown as one dimensional in Figure 1-1, it is a two dimensional array with rows and columns of detector-CCD elements. Now, we can limit the field of view of each element in a row so that each element only sees a portion of the scene. If we move the entire array upward by an amount equal to the element field of view the second row will see the same

portion of the scene that the first row saw. This same effect can be achieved by sweeping a mirror showing the scene vertically down the array. If the detected charge in row 1 elements is shifted down to row 2 at the same time that the identical portion of the scene is swept to row 2 elements (as shown in Figure 1-2), we can enhance the signal to noise ratio. The signal strengths add coherently and the noise adds incoherently in the same fashion as repeated radar returns on a target. The CCD portion of the array element is the mechanism for adding the signals and noise . This is called Time Delay Integration and has been referred to as signal processing on the focal plane. Note in Figure 1-2, that the image is shifted from the focal plane at the rate of one megapixel/sec but the PI/SO (parallel in/series out) shift register receives a line of image at a time.

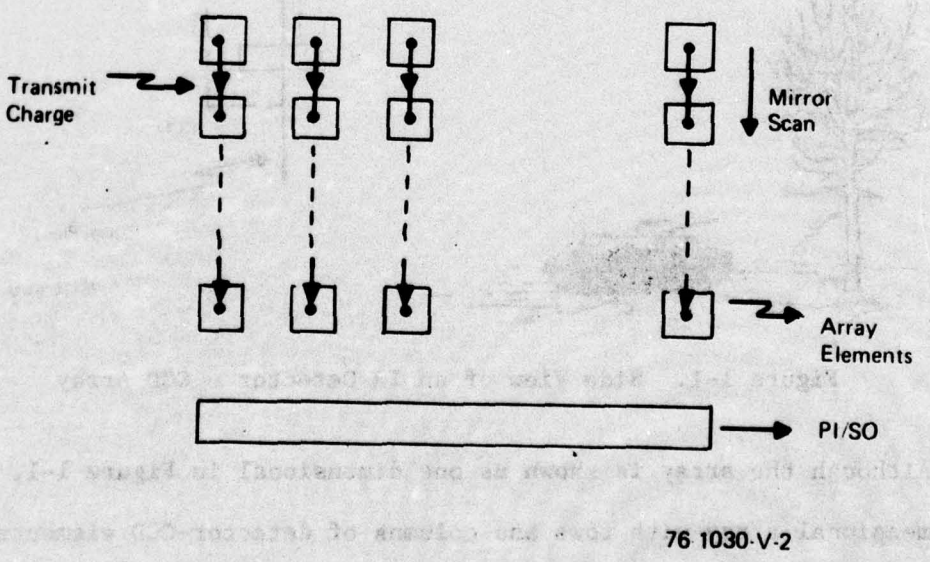


Figure 1-2 Time Delay Integration

For the purpose of this work we shall be enlarging the conventional definition of signal processing on the focal plane to include cueing functions, and there are several approaches to accommodating cueing functions on the focal plane. One way is to simply enlarge the focal plane chip and place several tiers of processing between the last row of TDI processing and the parallel in/series out shift register. Another possible approach is to perform cueing functions as the charges are being generated and passed down the array; here there is an opportunity to address the x-y position of a particular pixel, but the integration is interrupted. In this quarter, we used the first approach only because it allowed us to attack the mathematical and logic functions required by the cueing functions directly.

The number of cueing functions placed on the focal plane will depend on the tradeoff among the availability of space on the focal plane, yields for chips of small size and complexity, chip performance at cryogenic temperatures, and the effects of a number of leads penetrating the dewar.

2.0 SYSTEM FLOW

This section describes a preferred set of algorithms developed by Maryland which tentatively comprise the cueing system. A system flow chart is shown in Figure 2-1. A description of data flow and storage requirements is included.

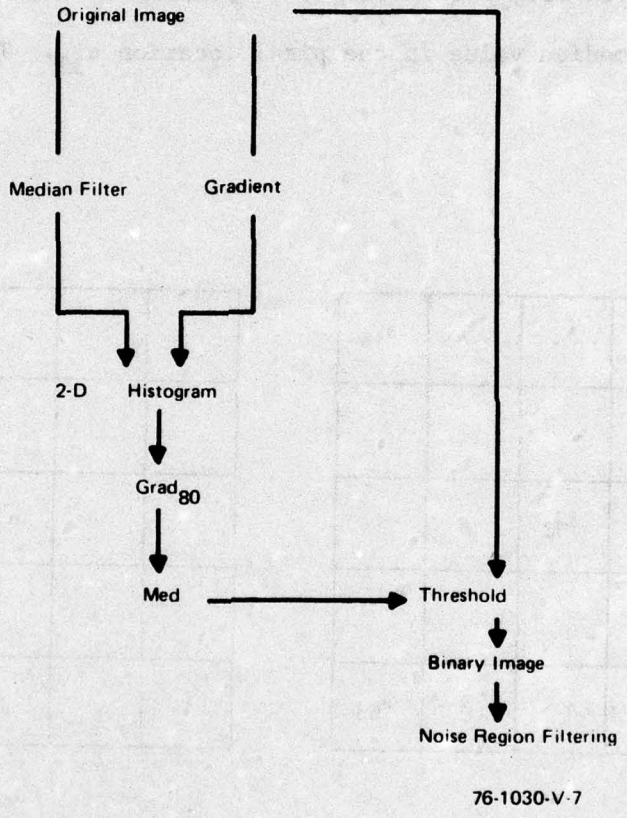


Figure 2-1. System Flow Chart

2.1 Algorithms

The order and a short description of each algorithm follows.

2.1.1 Median Filter

The Median Filter is the first algorithm performed and acts to extract the median gray level from a 5 x 5 array of pixels, as shown in Figure 2-2, and to place that median value in the pixel location a_{33}. The median value is

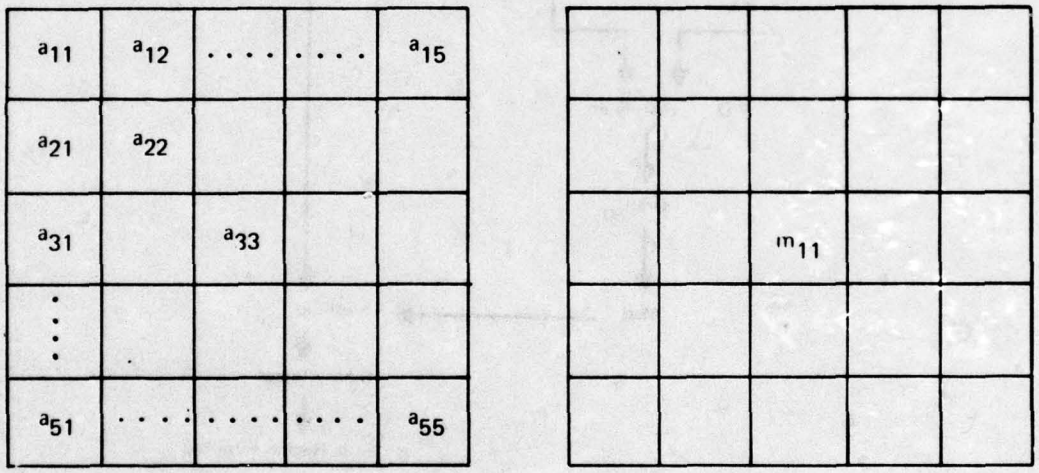


Figure 2-2. Median Filter

defined as the 13th gray level value in an ordering of the 25 gray levels by magnitude, counting from either end. The Median Filter acts as a moving 5 x 5 window in that having obtained m_{11}, column 1 (a_{11}, a_{21}, ... a_{51}) is removed from the ordering and column 6 (a_{16}, a_{26}, ... a_{56}) is added with the accompanying reordering and the 13th gray level value, m_{12}, is obtained, and placed in the a_{14} position. Having completed an entire line of image, row 1 is dropped, row 6 is added and the process is repeated. Maryland has chosen to ignore algorithm computations along the image edges for now.

2.1.2 Threshold Algorithm

The Threshold Algorithm first computes an operator, $OP = \max \{ |A-B|, |C-D| \}$ based on four regions A, B, C, and D, each of which consists of 4 x 4 pixels as shown in Figure 2-3 and the composite as seen in Figure 2-4.

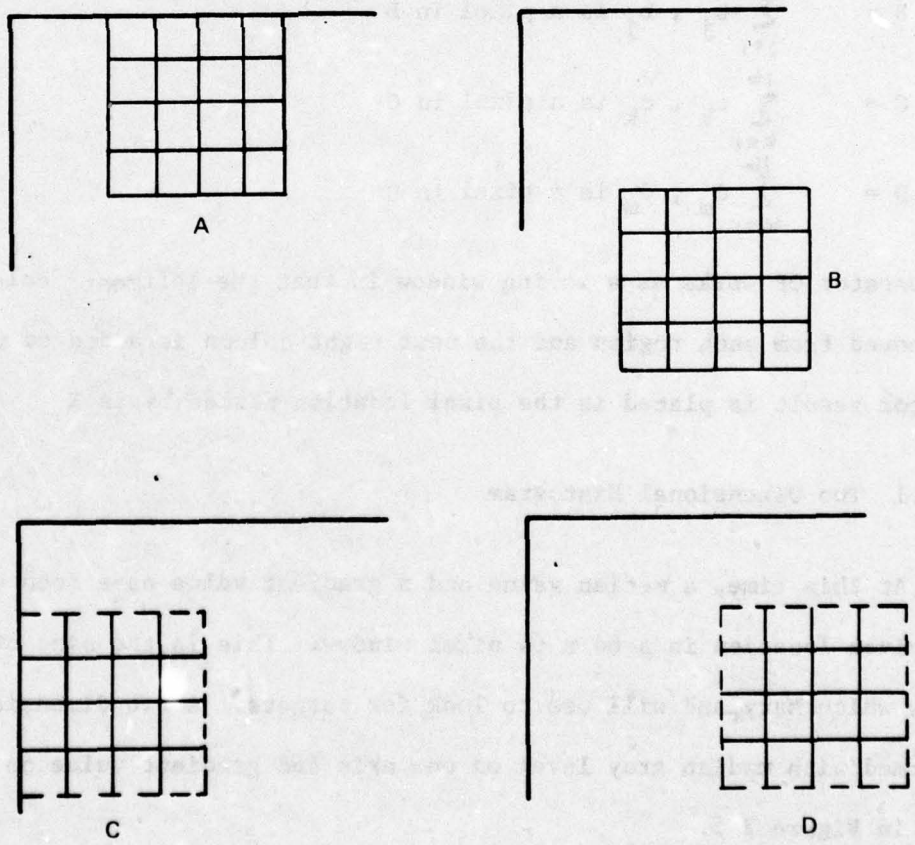
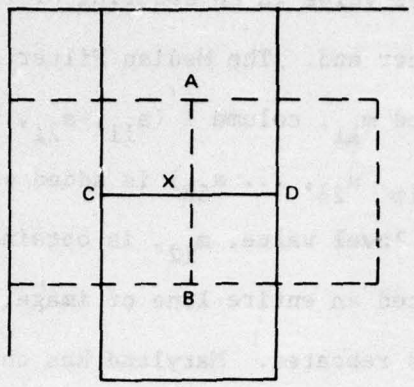


Figure 2-3. Regions A, B. C, and D

76-1030-V-4



76-1030-V 5

Figure 2-4. The Composite of A, B, C, and D

where:

$$A = \sum_{i=1}^{16} a_i \text{ , } a_i \text{ is a pixel in A}$$

$$B = \sum_{j=1}^{16} b_j \text{ , } b_j \text{ is a pixel in B}$$

$$C = \sum_{k=1}^{16} c_k \text{ , } c_k \text{ is a pixel in C}$$

$$D = \sum_{m=1}^{16} d_m \text{ , } d_m \text{ is a pixel in D}$$

The operator OP works as a moving window in that the leftmost column is removed from each region and the next right column is added to each. The operator result is placed in the pixel location marked by an X.

2.1.2.1 Two Dimensional Histogram

At this time, a median value and a gradient value have been computed for each pixel location in a 64 x 64 pixel window. This is the size of the moving window which Maryland will use to look for targets. A two dimensional histogram is formed with median gray level on one axis and gradient value on the other as shown in Figure 2-5.

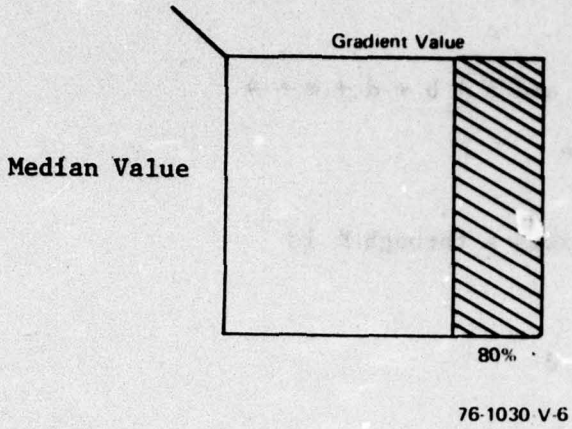


Figure 2-5. Histogram

Then a gradient value corresponding to (e.g.) 80% of the gradient population, $grad_{80}$, is found and the median values whose corresponding gradient value are greater than $grad_{80}$ are averaged. This subset of the histogram is shown as the cross hatched section in Figure 2-5. The average of this subset of median values, $\overline{med}$, is then used to threshold the original 64 x 64 image of gray level values.

2.1.2 Binary Operator

As stated, the average of the subset of median values is used to threshold the 64 x 64 image of original gray level values. Those gray level values which are equal to or exceed $\overline{med}$ are set equal to 1, and those values which are less than $\overline{med}$ are set equal to zero. The result is a binary image.

2.1.3 Noise Region Filtering

The Noise Region Filtering Algorithm acts to remove noise bursts which have survived the previous thresholding but are small in area and isolated. This algorithm is a shrink and expand process with the following logic. The shrink test may be stated as

$$q'(c) = \begin{cases} 1 \text{ if } c = 1 \text{ and } a + b + d + e = 4 \\ 0 \text{ otherwise} \end{cases}$$

where the arrangement of pixels a through e is

$$\begin{array}{ccc} & a & \\ b & c & d \\ & e & \end{array}$$

The expansion test may be stated as

$$q'(c) = \begin{cases} 1 \text{ if } c = 1 \text{ or } a + b + d + e > 0 \\ 0 \text{ otherwise} \end{cases}$$

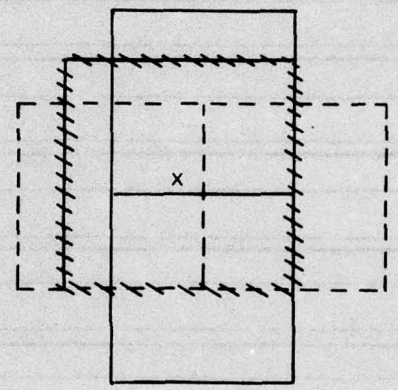
and the arrangement of the pixels is the same as above. Each test is repeated three times.

Having described each algorithm to a point necessary for hardware implementation, the next step is to see how the algorithms interface with each other and the focal plane structure.

2.2 Data Flow

2.2.1 Obtain Image Data for Median Filter and Gradient Operator

From preceding discussions of the Median Filter and Gradient Operator, it was seen that the Median Filter required 5 lines for processing. Similarly, the Gradient Operator required 8 lines. If we superimpose the Median Filter and the Gradient Operator so that their respective results would represent the same pixel location, the combination is seen in Figure 2-6.



76-1030-V-8

Figure 2-6. Super-Position of Median and Gradient Operators

It is noted from Figure 2-1 that the original image will be used later in a thresholding operation to produce a binary image; this means the 64 x 64 image must be stored somewhere while the Median Filter and Gradient Operator are working through the image. Coupled with the 8 line requirement for the Gradient Operator, a rough cut can be attempted at a focal plane configuration as seen in Figure 2-7.

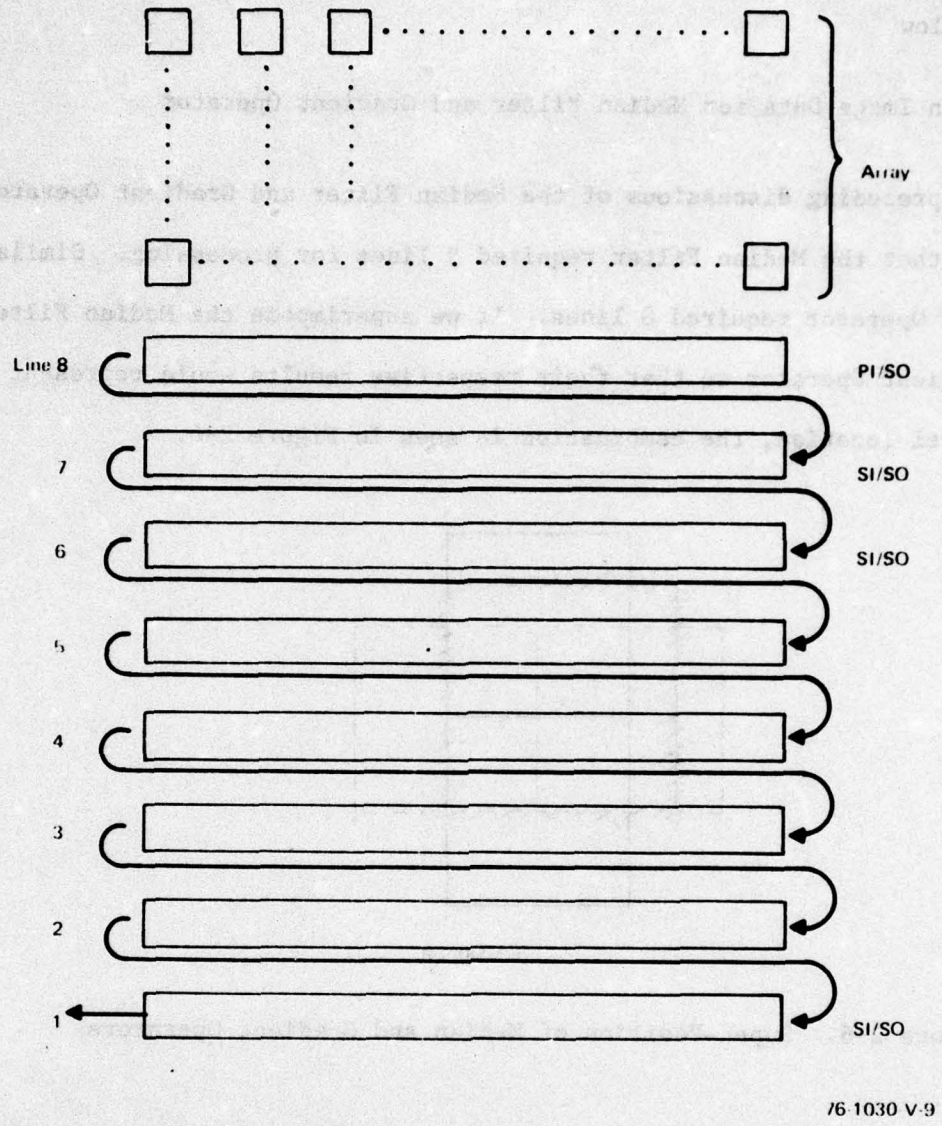


Figure 2-7. Focal Plane Configuration

The uppermost CCD register is a parallel in, series out (PI/SO) register, and the other CCD registers are all series in, series out (SI/SO) so that the 8 lines are delayed in a serpentine delay line. The output of the serpentine delay line would go to a storage memory where the original image is delayed. The next step is to examine how the individual elements will be nondestructively read from the circulating memory in order to form the Median Filter and the Gradient operator. Consider Figure 2-8, where the individual pixels are shown as located in the image (2-8a) and as found in the serpentine memory (2-8b).

The region A (4 x 4) is composed of

$$a_{13} \quad a_{14} \quad a_{15} \quad a_{16}$$

$$a_{23} \quad a_{24} \quad a_{25} \quad a_{26}$$

$$a_{33} \quad a_{34} \quad a_{35} \quad a_{36}$$

$$a_{43} \quad a_{44} \quad a_{45} \quad a_{46}$$

and the sum, $\overline{A}$, is composed of all 16 terms. These 16 terms can be obtained from the serpentine memory by nondestructive readout from these positions as shown in Figure 2-8. Similarly, the region C (4 x 4) is composed of

$$a_{31} \quad a_{32} \quad a_{33} \quad a_{34}$$

$$a_{41} \quad a_{42} \quad a_{43} \quad a_{44}$$

$$a_{51} \quad a_{52} \quad a_{53} \quad a_{54}$$

$$a_{61} \quad a_{62} \quad a_{63} \quad a_{64}$$

and the sum, $\overline{C}$, is composed of all 16 terms. These terms are also obtained from the serpentine memory by nondestructive readout from the positions shown in Figure 2-8. The nondestructive readout positions are also shown for the sums $\overline{B}$ and $\overline{D}$.

The Median Filter calculation requires the block of numbers

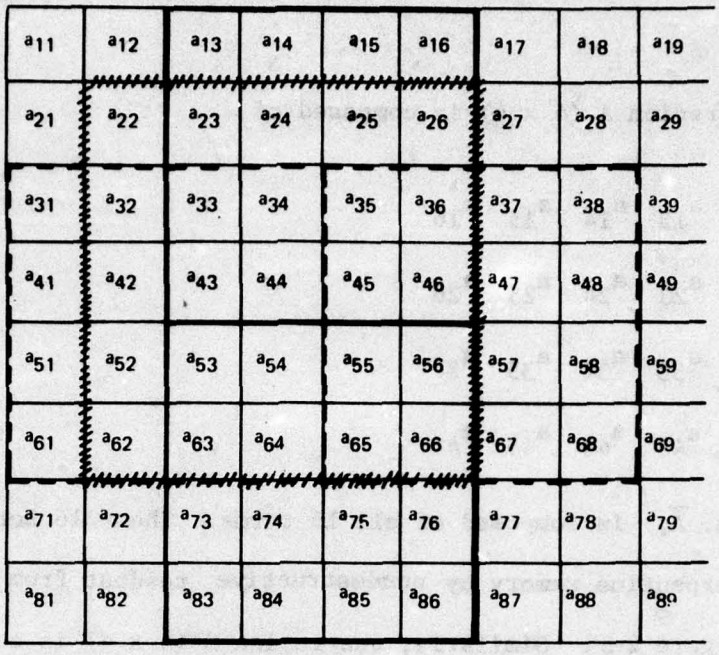
$$a_{22} \quad a_{23} \quad a_{24} \quad a_{25} \quad a_{26}$$

$$a_{32} \quad a_{33} \quad a_{34} \quad a_{35} \quad a_{36}$$

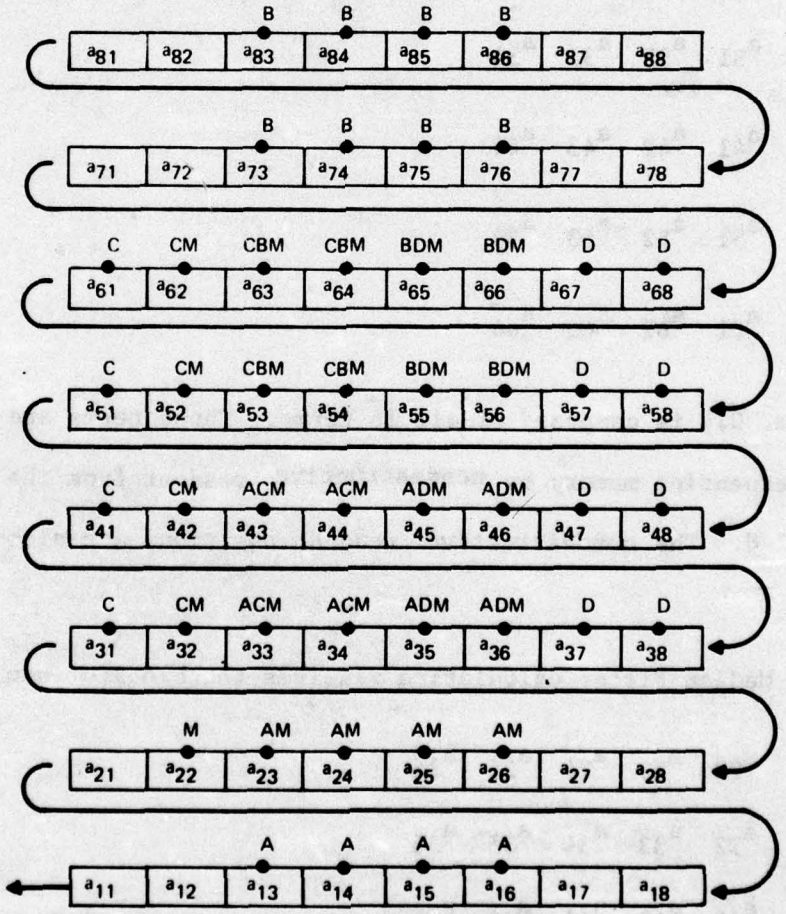
$$a_{42} \quad a_{43} \quad a_{44} \quad a_{45} \quad a_{46}$$

$$a_{52} \quad a_{53} \quad a_{54} \quad a_{55} \quad a_{56}$$

$$a_{62} \quad a_{63} \quad a_{64} \quad a_{65} \quad a_{66}$$



2-8a. Pixel Locations in the Image



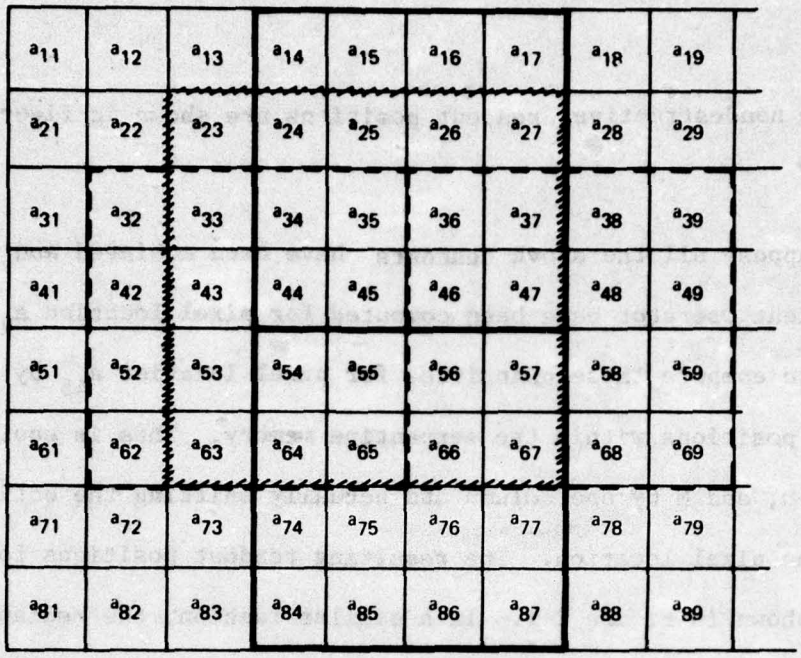
2-8b. Pixel Locations in Serpentine Memory 6 1030-V-10

Figure 2-8. Focal Plane Registers

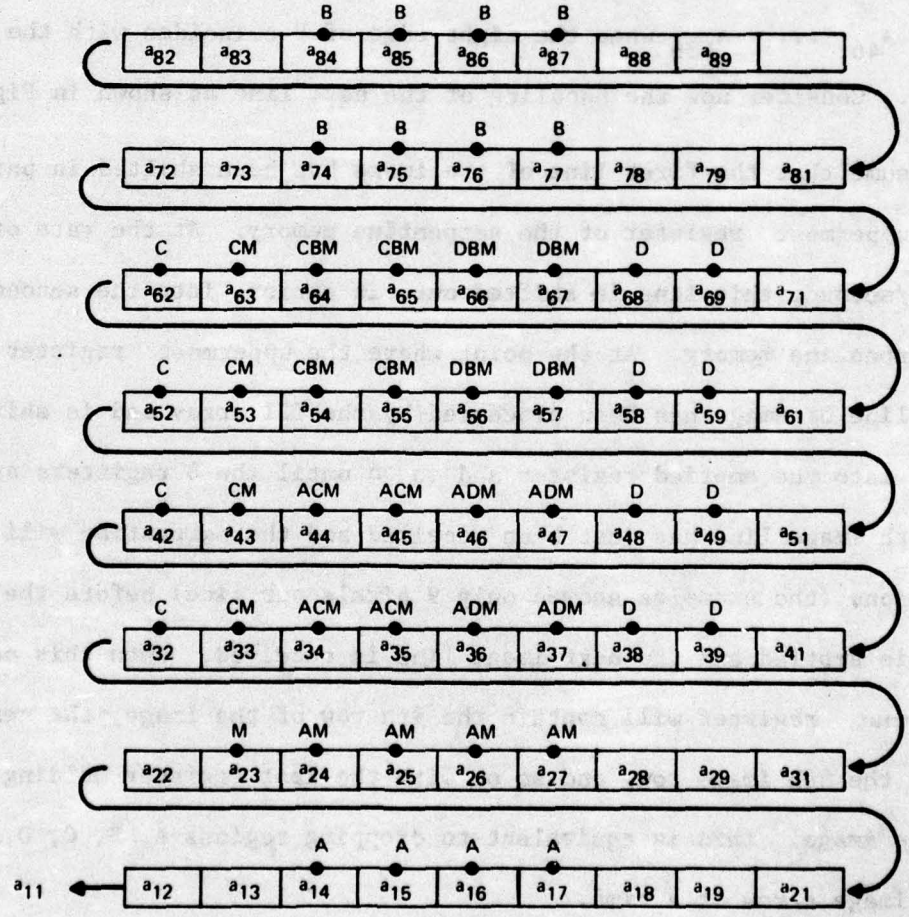
and their nondestructive readout positions are shown in Figure 2-8 and designated by M.

Suppose all the above readouts have been achieved and the Median Filter and Gradient Operator have been computed for pixel location a_{44}. It is now desired to compute these quantities for pixel location a_{45} by using the same readout positions within the serpentine memory. This is equivalent to shifting A, B, C, D, and M by one column and actually shifting the entire serpentine memory one pixel location. The resulting readout positions for pixel location a_{45} are shown in Figure 2-9. In a similar fashion, the Median Filter and Gradient Operator can be computed for the pixel locations in the 4th row, i.e., a_{44}, a_{45}, a_{46}, ... a_{636} when the right edge of D coincides with the edge of the image. Consider now the handling of the next line as shown in Figure 2-10.

Assume that the first line of the image has been shifted in parallel into the uppermost register of the serpentine memory. At the rate of 1 megapixel/second, this line is shifted out, in series, into the second register of the serpentine memory. At the point where the uppermost register is empty, the next line of image has been processed by the TDI array and is shifted, in parallel, into the emptied register and so on until the 8 registers are filled. Now the 8th image line has just been received and the serpentine will shift 640 positions (the examples showed only 9 pixels per line) before the upper register is emptied and the next image line is received. When this occurs, the uppermost register will contain the 9th row of the image, the next register will hold the 8th image row, and so on with the last register holding the 2nd row of the image. This is equivalent to dropping regions A, B, C, D, and M down the image a row at a time.



2-9a. Pixel Locations in the Image



2-9b. Pixel Locations in Serpentine Memory 76-1030-V-11

Figure 2-9. Readout Pixels for a_{45} Element

a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}	a_{29}
a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	a_{36}	a_{37}	a_{38}	a_{39}
a_{41}	a_{42}	a_{43}	a_{44}	a_{45}	a_{46}	a_{47}	a_{48}	a_{49}
a_{51}	a_{52}	a_{53}	a_{54}	a_{55}	a_{56}	a_{57}	a_{58}	a_{59}
a_{61}	a_{62}	a_{63}	a_{64}	a_{65}	a_{66}	a_{67}	a_{68}	a_{69}
a_{71}	a_{72}	a_{73}	a_{74}	a_{75}	a_{76}	a_{77}	a_{78}	a_{79}
a_{81}	a_{82}	a_{83}	a_{84}	a_{85}	a_{86}	a_{87}	a_{88}	a_{89}
a_{91}	a_{92}	a_{93}	a_{94}	a_{95}	a_{96}	a_{97}	a_{98}	a_{99}

76-1030-V-12

Figure 2-10. A, B. C. D Regions for Line 6

In summary, we have described how the appropriate sets of pixels forming A, B, C, D, and M regions of the image are obtained from the image while still preserving the image for additional processing later, see Figure 2-11. We reserve for a later section the actual computation of the Median Filter and the Gradient Operator, and next consider a parallel implementation.

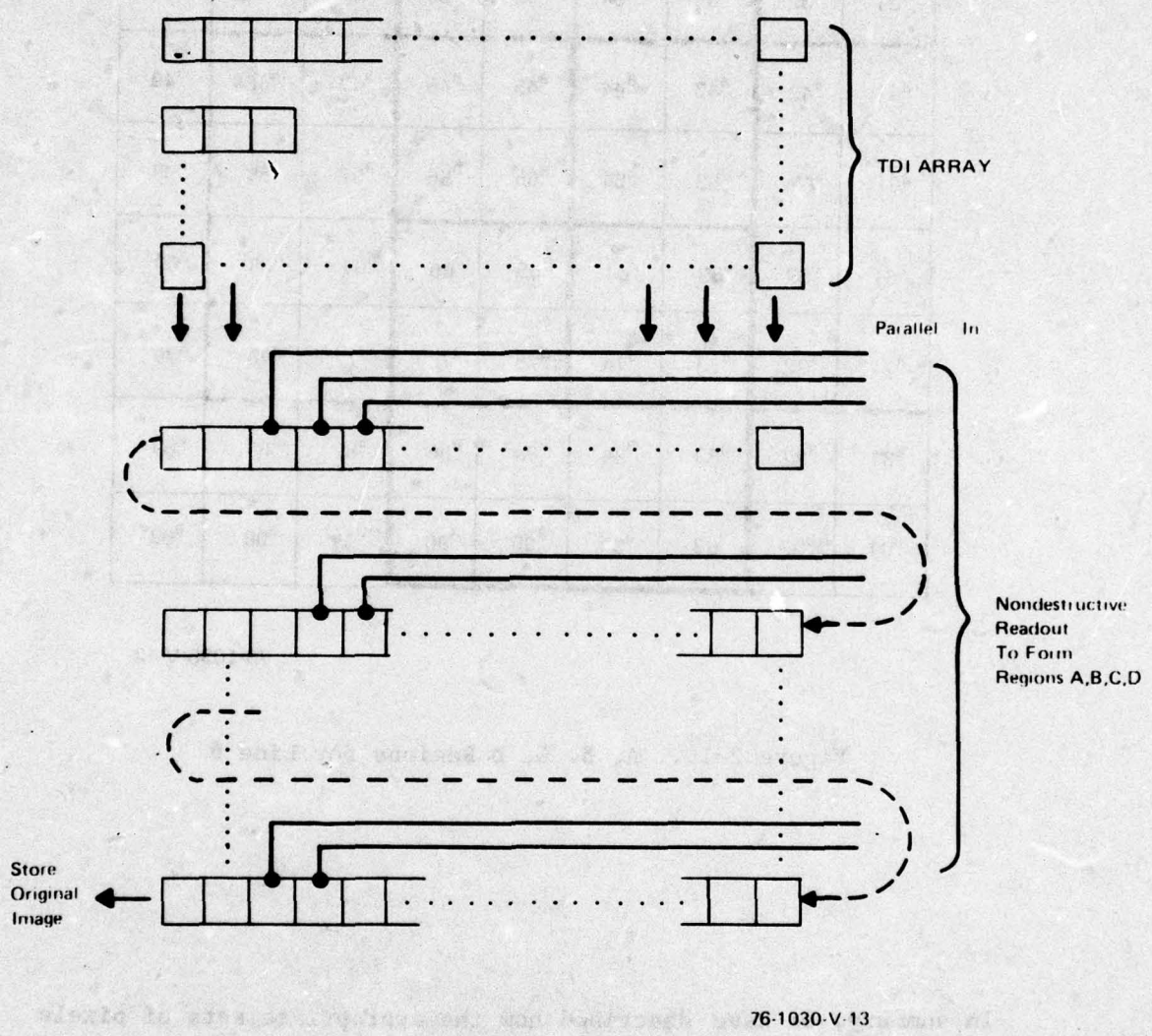


Figure 2-11. Details of Serpentine Delay Line

2.2.2 Parallel Extraction of Median Filter and Gradient Operator

In the previous discussions, we have implicitly assumed that a single Median Filter and Gradient Operator device would process the data. This may not be the case because either the CCD implementation is not capable of 1 megapixel/sec. or a higher data rate, e.g., 10 megapixels/sec. is desired. It is appropriate to consider a simple example of a parallel configuration.

The image comes into the serpentine memory a line at a time. Assuming there are 640 pixels per line, ten windows can be formed simultaneously (Maryland is working with a 64 x 64 window at present). The sensor array need not be divided into ten sections, but the PI/SO register is divided. Each section may have its own serpentine memory and certainly it will have its own Median Filter and Gradient Operator. Figure 2-12 shows such an arrangement. The memory used to hold the original image would be divided in a similar fashion as shown in Figure 2-13 to maintain the data rates.

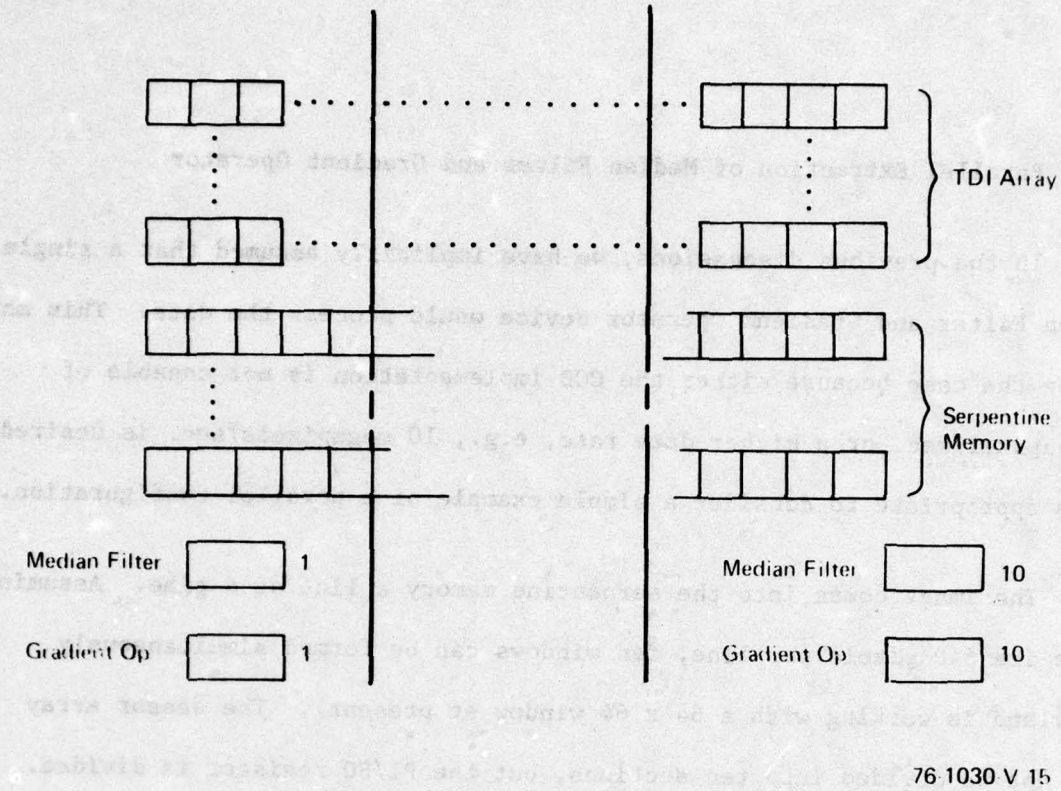


Figure 2-12. Parallel Organization of Focal Plane

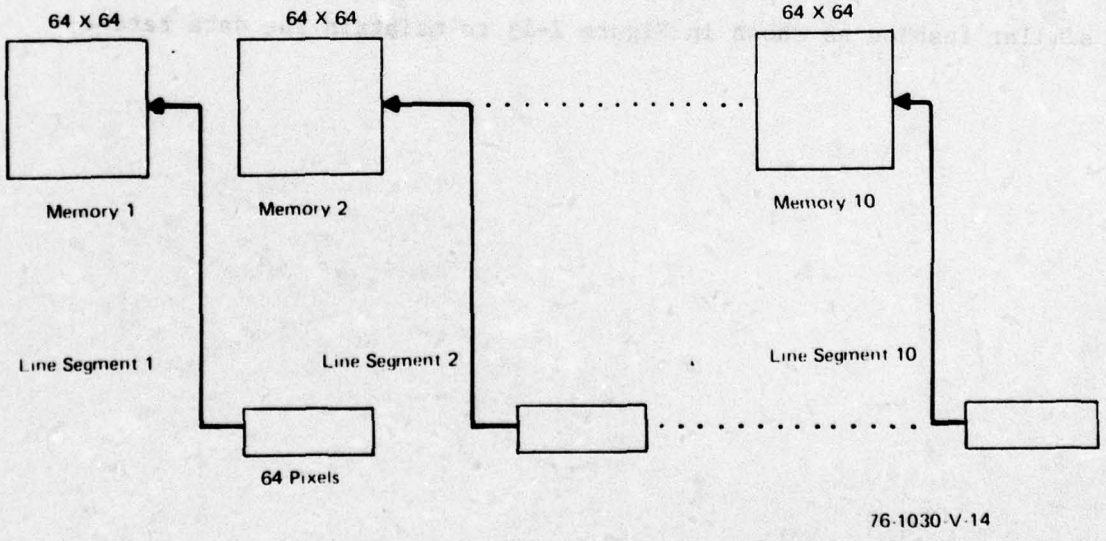


Figure 2-13. Memory Organization for Original Image

3.0 Hardware Implementation

In the prior sections, we have defined focal plane processing and discussed the data flow through the system, thereby gaining an idea of the storage data handling and speed requirements of the system. In this section, we shall discuss specific hardware techniques to obtain the data handling required and perf the algorithms.

3.1 Serpentine Delay Line

The formation of the serpentine delay line and the nondestructive readout features as implemented in CCD technology were discussed in Section 4.5 of the First Quarterly Report.

3.2 Threshold Algorithm

3.2.1 Gradient Operator

The Gradient Operator as defined by Maryland is $G = \max \{|A-B|, |C-D|\}$ where A, B, C, D are each the sum of 16 pixel gray levels whose geometrical arrangement was described in Sections 2.1.2 and 2.2 of this report. A, B, C, and D can be redefined as

$$A_{ij} = \sum_{m=j}^{j+2} \sum_{k=1}^{i+2} a_{ij} \qquad C_{ij} = \sum_{m=j-1}^{j+1} \sum_{k=i+1}^{i+3} a_{ij}$$

$$B_{ij} = \sum_{m=j}^{j+2} \sum_{k=i+2}^{i=4} a_{ij} \qquad D_{ij} = \sum_{m=j+1}^{j+3} \sum_{k=i+1}^{i+3} a_{ij}$$

In Section 2.2.1, we discussed how the appropriate a_{ij}'s for each sum are obtained from the serpentine memory. Further, the CCD technology to non-destructively readout and form sums was discussed in Section 4.5 of the First Quarterly Report. Then, the focal plane is as shown in Figure 3-1,

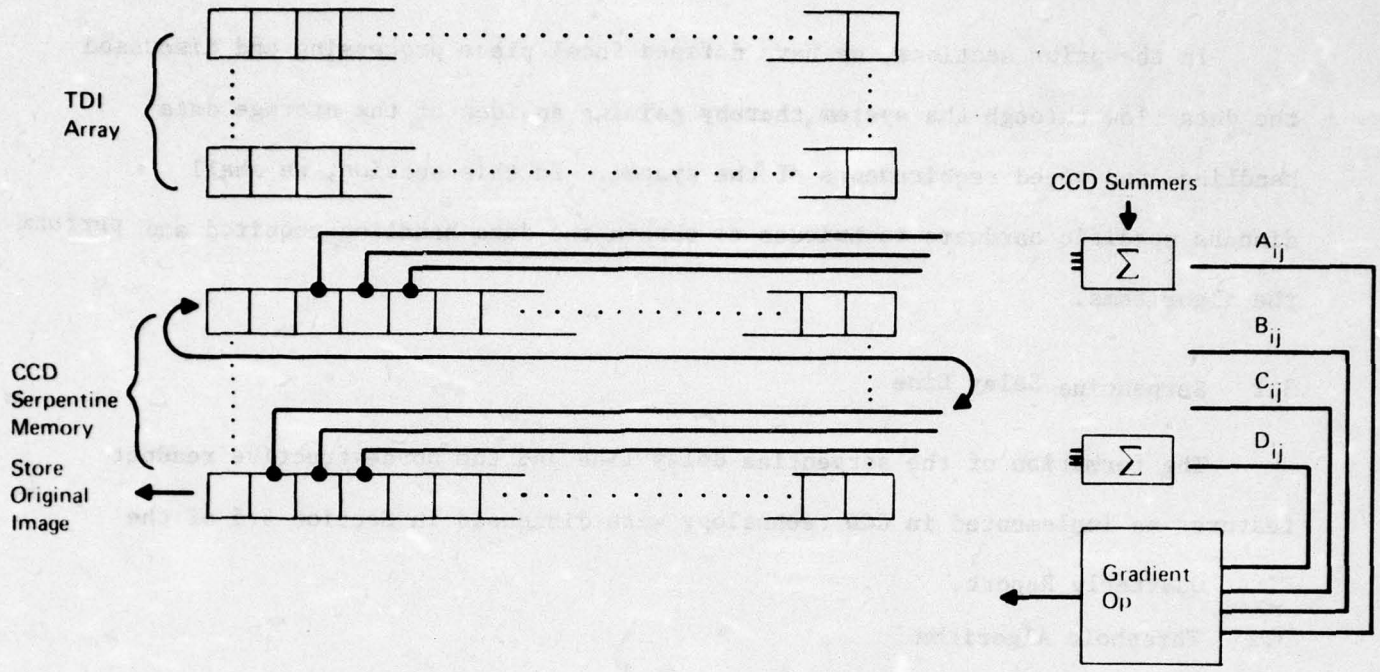


Figure 3-1. Focal Plane and Gradient Operator

and we shall be concentrating on the contents of the box labelled Gradient Operator. Utilizing our redefinition of A, B, C, and D as A_{ij}, B_{ij}, C_{ij} and D_{ij}, the Gradient Operator may be redefined as

$$G_{ij} = \begin{cases} |A_{ij} - B_{ij}| & \text{if } |A_{ij} - B_{ij}| > |C_{ij} - D_{ij}| \\ |C_{ij} - D_{ij}| & \text{if } |C_{ij} - D_{ij}| \geq |A_{ij} - B_{ij}|. \end{cases}$$

The key CCD technique to achieve G_{ij} is subtraction using the potential Equilibration Method which does not correspond to conventional subtraction.

Consider the structure in Figure 3-2 with the two signal inputs A_{ij}, B_{ij} and the diffusion diode input D_{in}.

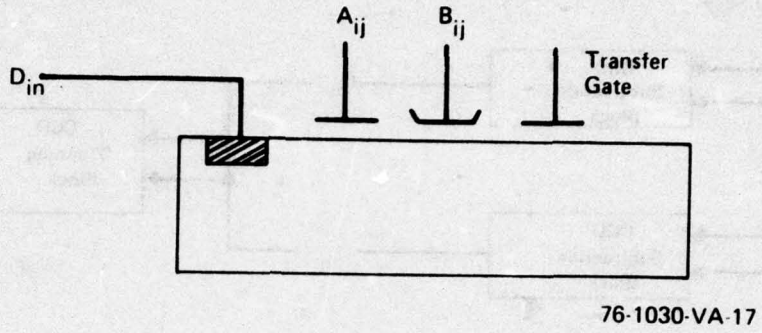


Figure 3-2. CCD Subtraction Block

Using potential equilibration, the charge injected via D_{in} and retained in the CCD well under the gate B_{ij} will not equal zero only if $|B_{ij}|$ is greater than $|A_{ij}|$. Mathematically, the amount of charge Q_{ij} produced in the CCD by subtracting A_{ij} from B_{ij} can be expressed as follows:

$$Q_{ij} = \begin{cases} K\,(B_{ij} - A_{ij}) & \text{for } |A_{ij}| < |B_{ij}| \\ 0 & \text{for } |B_{ij}| < |A_{ij}| \end{cases}$$

where K is a constant. The aforementioned subtraction operation can be represented by a module shown in Figure 3-3.

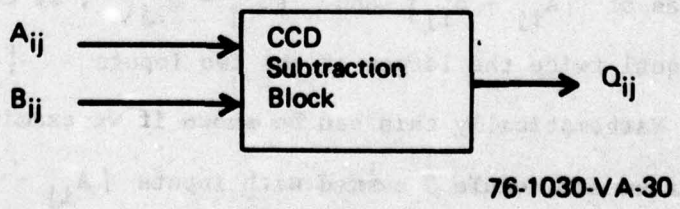


Figure 3-3. CCD Subtraction Module

If two of the modules in Figure 3-3 are placed in parallel and the inputs (A_{ij}, B_{ij}) to each are commuted, one of the outputs will not be equal to zero, as in Figure 3-4.

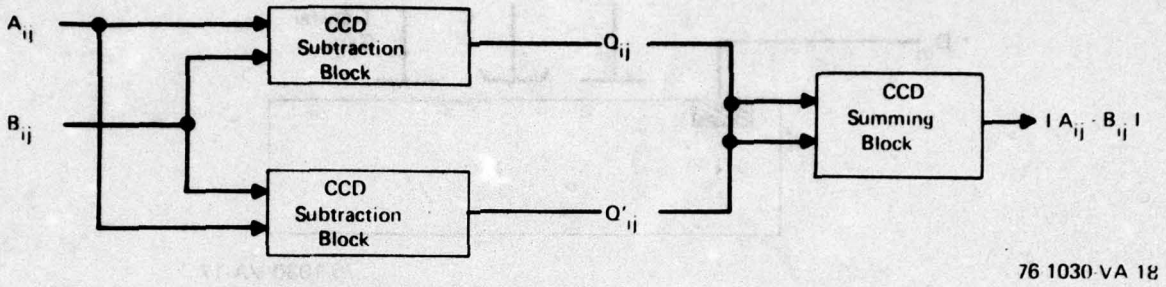


Figure 3-4. CCD Absolute Value Operator

A sum of the two outputs Q_{ij} and Q'_{ij} will yield the absolute value of $A_{ij} - B_{ij}$. A similar operation performed on C_{ij} and D_{ij} will yield the absolute value of the difference between C_{ij} and D_{ij}.

The largest absolute difference, i.e., $|A_{ij} - B_{ij}|$ or $|C_{ij} - D_{ij}|$, can be obtained if we make use of a CCD non-destructive readout module (CCD NDRO) and form an assembly shown in Figure 3-5. The modules labelled by ABS represent the diagram shown in Figure 3-4. The output from ABS module 3 will be equal to $(|A_{ij} - B_{ij}| - |C_{ij} - D_{ij}|)$ or $(|C_{ij} - D_{ij}| - |A_{ij} - B_{ij}|)$ depending on which quantity is positive. If we add to the output of ABS module 3 the values of $|A_{ij} - B_{ij}|$ and $|C_{ij} - D_{ij}|$, it can be shown that the output will equal twice the larger of the two inputs $|A_{ij} - B_{ij}|$ and $|C_{ij} - D_{ij}|$. Mathematically this can be shown if we examine all the possible outputs from ABS module 3 summed with inputs $|A_{ij} - B_{ij}|$ and $|C_{ij} - D_{ij}|$. The results are shown in Figure 3-6, and illustrate that the output

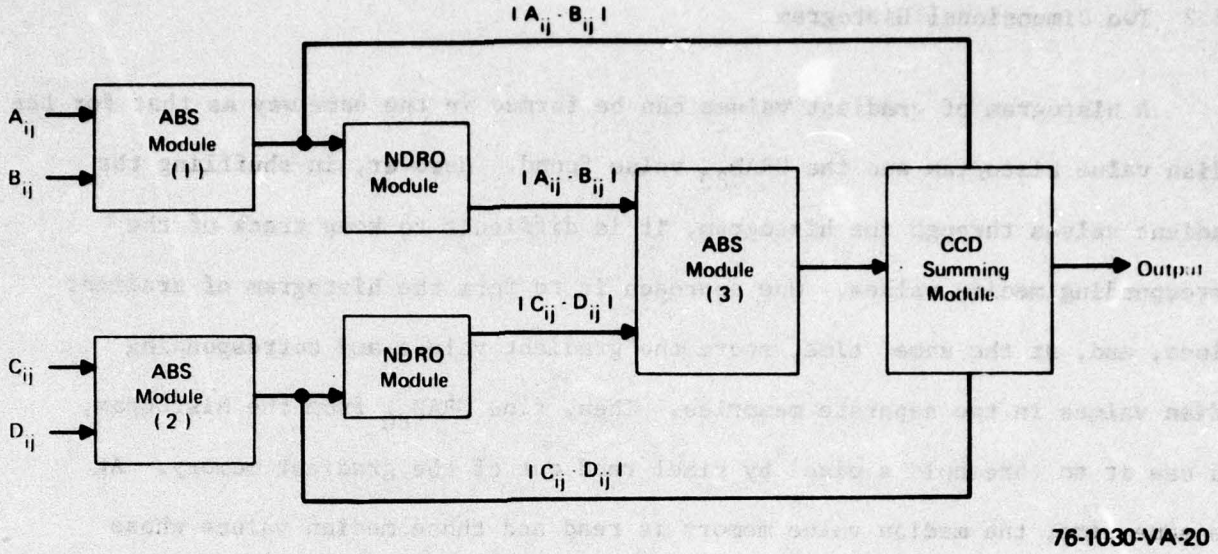


Figure 3-5. CCD Gradient Operator

of the CCD Gradient Operator is indeed equal to the largest input $|A_{ij} - B_{ij}|$ or $|C_{ij} - D_{ij}|$.

Condition	$\|A_{ij}-B_{ij}\| < \|C_{ij} - D_{ij}\|$	$\|C_{ij} - D_{ij}\| < \|A_{ij} - B_{ij}\|$
ABS Module 3 Output	$\|C_{ij} - D_{ij}\| - \|A_{ij} - B_{ij}\|$	$\|A_{ij} - B_{ij}\| - \|C_{ij} - D_{ij}\|$
Summing CCD Inputs	$\|C_{ij} - D_{ij}\| + \|A_{ij} - B_{ij}\|$	$\|C_{ij} - D_{ij}\| - \|A_{ij} - B_{ij}\|$
Outputs From Summing CCD	$2 \|C_{ij} - D_{ij}\|$	$2 \|A_{ij} - B_{ij}\|$

76-1030-VA-29

Figure 3-6. Outputs

3.2.2 Two Dimensional Histogram

A histogram of gradient values can be formed in the same way as that for the median value histogram and the $GRAD_{80}$ value found. However, in shuffling the gradient values through the histogram, it is difficult to keep track of the corresponding median values. One approach is to form the histogram of gradient values, and, at the same time, store the gradient values and corresponding median values in two separate memories. Then, find $GRAD_{80}$ from the histogram, and use it to threshold a pixel by pixel read out of the gradient memory. At the same time, the median value memory is read and those median values whose corresponding gradient exceed $GRAD_{80}$ are kept and averaged as in Figure 3-7. This approach implies a storage capacity for a histogram and two images which are each 64 x 64.

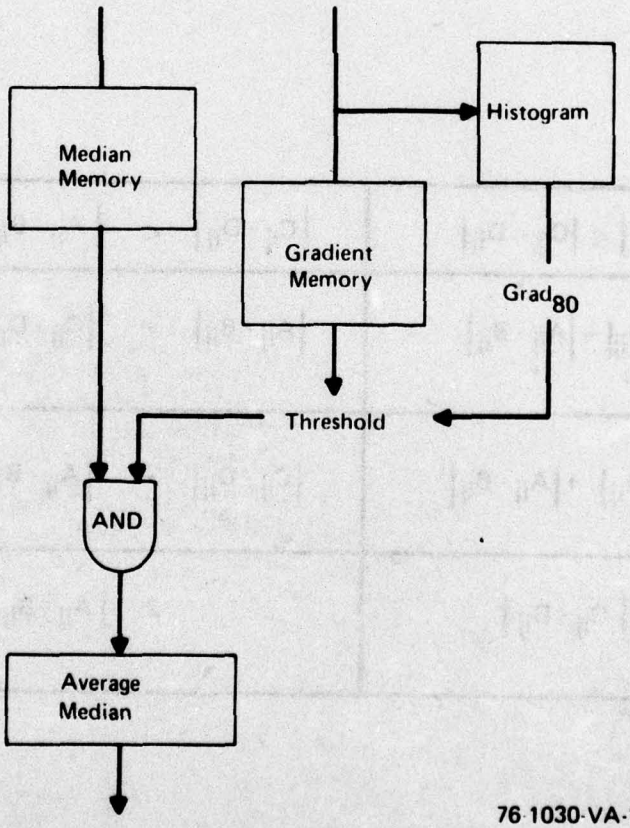


Figure 3-7. Flow of Two Dimensional Histogram

Another approach involves a RAM structure to form the two dimensional histogram. Recall that the median values have been converted to binary and when the gradient histogram is formed, the gradient values will also be binary numbers. Suppose that they will each be represented by 5 bits so that each varies from 0 to 31. If the gradient and median values are catenated to form a single 10 bit address, the range of possible addresses is 1024, and the RAM is 1024 addresses long. Suppose further that the maximum number of occupants of each gradient median value location is 64, then the RAM is 1024 x 6 bits wide. In constructing a two dimensional histogram, the RAM must be read, its contents incremented by 1, and then written back into the same location. Here we are talking about more than one RAM and possibly a cluster of 4-6 to obtain length and width, each RAM at least a 22 pin pack which is incompatible with focal plane processing.

3.3 Median Filter

In order to obtain the median value from a set of numbers, it is necessary to obtain an ordering of their numerical magnitudes. This, of course, is equivalent to a histogram. One approach to this problem of forming a histogram is to convert each analog number to a binary one but represent it by a "thermometer code", i.e., the charge level corresponding to a gray scale pixel magnitude is represented by a vertical column of 1's as shown in Figure 3-8.

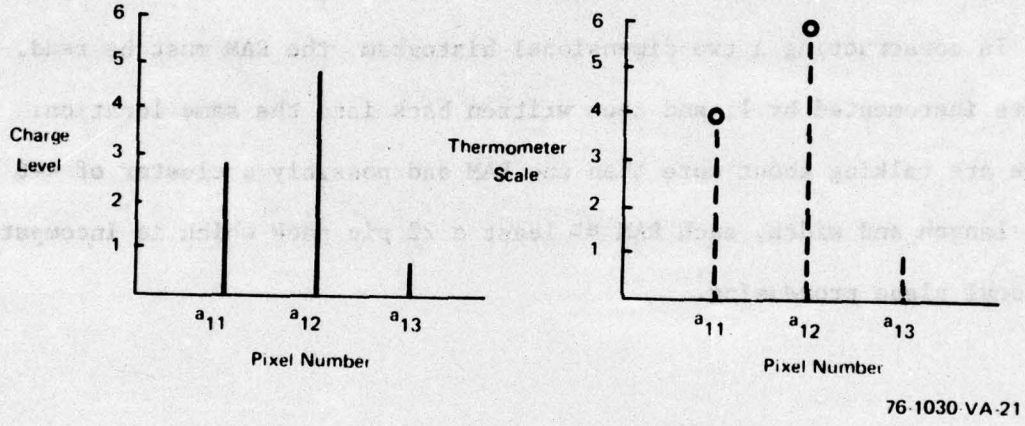


Figure 3-8. Thermometer Scale

Before discussing the mechanics of forming the histogram, the first order of business is the CCD A/D converter.

3.3.1 CCD A/D Converter

This CCD structure will show how an A/D operation can be achieved on the IR focal plane at cryogenic temperatures. The key element in the A/D converter is a CCD which exhibits very low voltage offsets. Such a device is needed to generate the different voltage levels required to form parallel mode A/D converters. These CCD input voltage levels cannot exhibit DC offset

nonuniformities. These are found in conventional CCD's and are attributed to threshold nonuniformities. Previous work in this area involved a parallel in/ series out CCD with parallel threshold variation. This CCD employing compensated direct injection (CDI) can be employed for A/D operations according to the following procedure. Each ith parallel input of the CDI-CCD can be thought of as a common gate MOS FET with a virtual drain as seen in Figure 3-9. The net current entering the CCD ith input node $I_{CCD}(i)$ is equal to the difference between $I_R(i) - I_S = I_{CCD}(i)$ only if the ith reference current $I_R(i)$ is larger in magnitude than the signal current I_S.

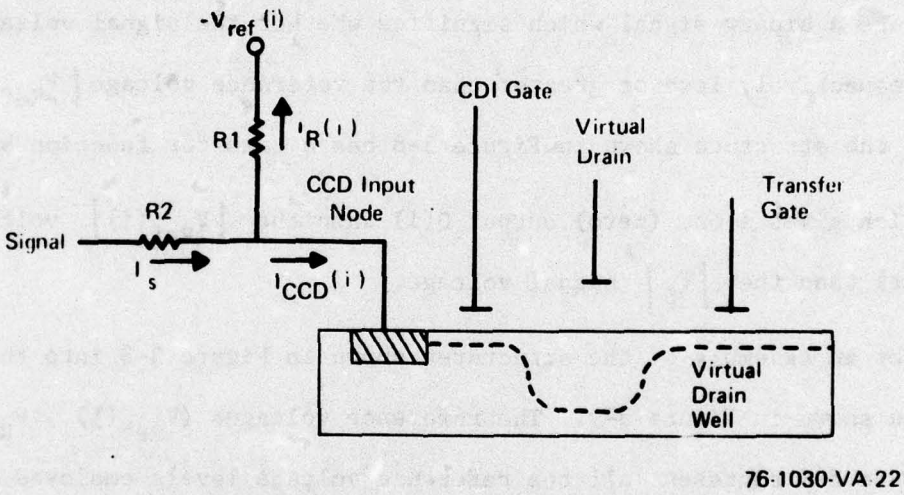


Figure 3-9. CCD Input Structure

Whenever $I_R(i)$ is less than I_S the current $I_{CCD}(i)$ entering the CCD is zero. Mathematically the amount of current entering the CCD virtual drain's well as a function of $I_R(i)$ and I_S can be expressed as

$$I_{CCD}(i) = \begin{cases} -I_R(i) + I_S & \text{for } (-I_R(i) + I_S) > 0 \\ 0 & \text{for } (-I_R(i) + I_S) \leq 0. \end{cases}$$

The amount of charge in any given CCD virtual drain will equal the product of the integration time T and the current value defined by the equation for $I_{CCD}(i)$.

Given sufficient integration time (T), the virtual drain's well will be completely full of charge or empty depending on the external conditions operated on by the above equations. Overflow of the accumulating charge in any one of the $\underline{i}$ virtual drains of the A/D converter can be prevented if the draining port adjacent to each virtual drain is incorporated into the A/D structure. The draining port can be made from a single gate and a drain diffusion bus. The potential on the overflow gate can be adjusted to a level equal to the charge potential in a filled virtual drain. Any additional charge entering the virtual drain will spill into the drain bus via the channel created by the gate of the overflow structure. The presence or absence of charge in the virtual drain's well represents a binary signal which signifies whether the signal voltage $|V_S|$ is respectively less or greater than the reference voltage $|V_{Ref}(i)|$. Summarizing, the structure shown in Figure 3-8 has a transfer function binary in nature which gives a one (zero) output Q(i) when the $|V_{Ref}(i)|$ voltage is less (greater) than the $|V_S|$ signal voltage.

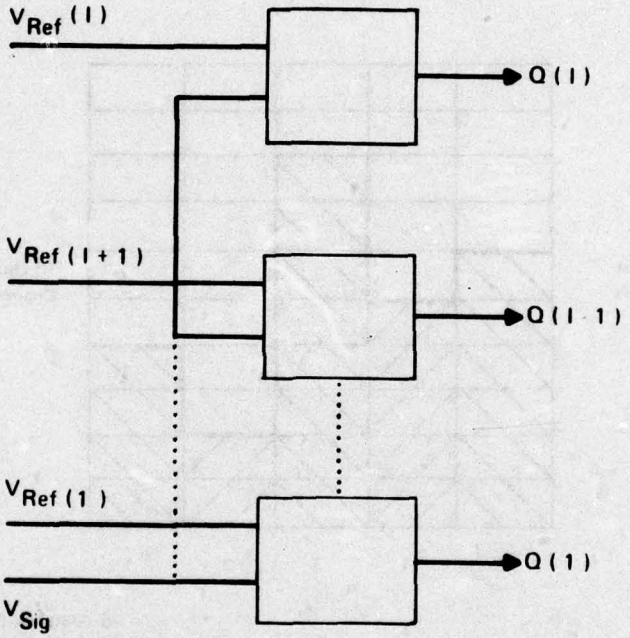
Consider an assembly of the structures shown in Figure 3-8 into the configuration shown in Figure 3-9. The reference voltages ($V_{Ref}(1)...V_{Ref}(I)$) shown in Figure 3-9 represent all the reference voltage levels employed by a parallel A/D converter. The number and values of the reference voltages $V_{Ref}(i)$ can be related to the A/D maximum input voltage (V_{Max}) and its least significant quantum level. The number of reference voltages required is equal to $I = V_{Max}/V_{Ref}(1)$. The relationship between different reference voltages can be expressed as

$$V_{Ref}(i) = V_{Ref}(i-k) + K\, V_{Ref}(i)$$

where $i-k \geq 1$ and $i \leq I$. For a given input signal (V_S) into the A/D converter (where $|V_{Ref}(i)| < |V_S| < |V_{Ref}(i+1)|$) the output (Q(i)) of the A/D converter shown in Figure 3-9 will be as follows.

$$Q(i) = \begin{cases} 0 \text{ for } K+1 \leq i \leq I \\ 1 \text{ for } i < K+1 \end{cases}$$

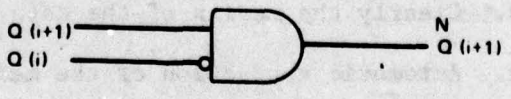
Such an output code for the analog input V_S is commonly referred to as a thermometer code. This thermometer code output can be changed into a code $\tilde{Q}(i)$ where only the $\tilde{Q}(i)$ equal to 1 with the lowest index i in the thermometer code will remain a 1 (i.e., $\tilde{Q}(i) = Q(i)$ and all other Q(K) will be equal to zero for $K \neq i$. Such a transformation from the thermometer code to the Q(i) code can be implemented by a group of logic blocks.



76-1030-VA-23

Figure 3-10. Thermometer Code

Each logic element will have two inputs and a single output as shown in Figure 3-11.



76-1030-VA-24

Figure 3-11. Logic Element

The logic block is simply two input AND gate where only one input is inverted. A group of these logic elements can readily convert the Q(i) to the $\tilde{Q}(i)$. The $\tilde{Q}(i)$ can be converted into a BCD code by conventional methods. The next subject of discussion in the Median Filter is formation of the Histogram.

3.3.2 Histogram

A signal processing structure will be discussed for extracting different threshold levels from a data ensemble. The proposed approach is easily adaptable for extraction of a median from a data ensemble. Moreover, with some modifications, a time varying median of N data elements located in a moving window can be extracted. Consider five data elements (see Figure 3-12) represented by a ten level thermometer code stored in a bank of Shift Register (S/R) CCD registers.

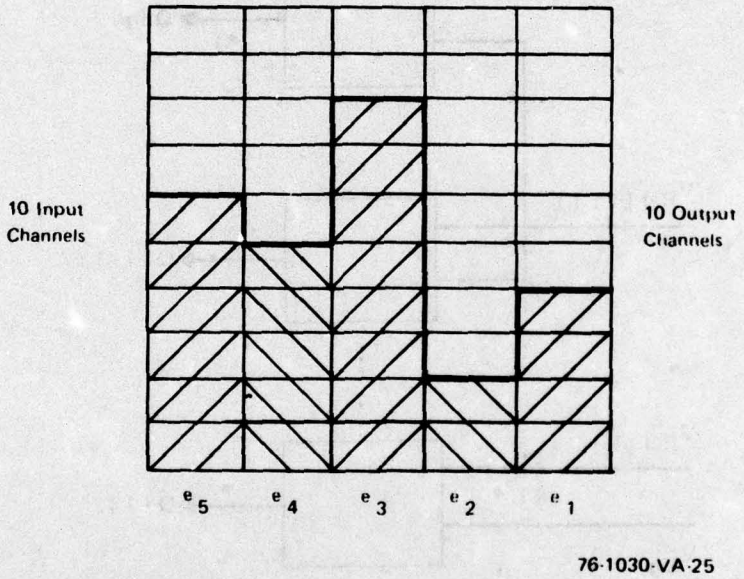


Figure 3-12. Storage Block for 5 Data Elements (e_1, e_2, e_3, e_4, and e_5)

The illustrated bank of S/R or CCD (i.e., memory bank) has ten parallel inputs and ten parallel outputs. Clearly the median of the data elements e_1, e_2, e_3, e_4, e_5 is the e_4 element. Automatic extraction of the median can be achieved if the five elements shown in Figure 3-12 are first arranged according to size and the middle element is read out from the median extracting module. Mechanization of median extraction can be readily achieved if the clock for the R^{th}

row of the S/R or CCD bank (shown in Figure 3-12) is slaved via a logic function to the signal appearing at the $\underline{R}^{th}$ input channel. In Figure 3-13 the clock of the memory module is slaved to the input signal via an AND gate. If the input signal at the input channel is a <u>one</u>, the clock to the Memory Module is enabled and a <u>one</u> will

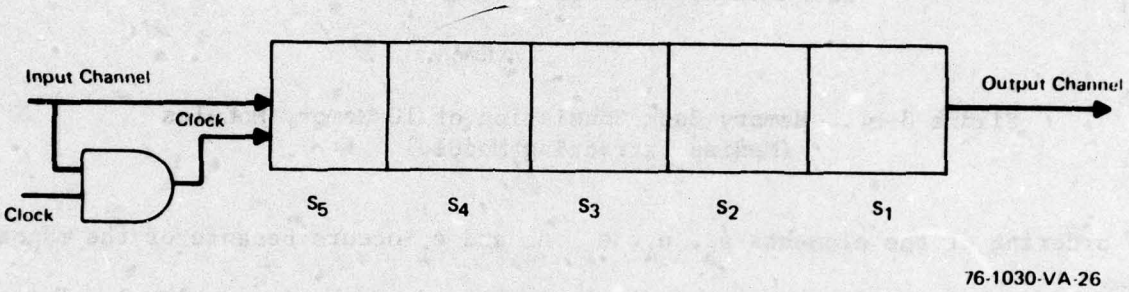


Figure 3-13. Clock to Memory Module (MM) Slaved by an AND Gate to Input Signal.

be clocked into the memory. Otherwise (i.e., when the input is a zero) the clock driving the memory module will be disabled and no information will be clocked into the memory module. Under the clock constraints outlined for the memory module (see Figure 3-13) a <u>one</u> signal will appear in any S_i location (i.e., S_5, S_4, S_3, S_2, S_1) only if all the preceding locations with a subscript larger than <u>i</u> are filled with a <u>one</u> signal.

Consider ten memory modules shown in Figure 3-12 assembled into a memory bank (see Figure 3-14) equal in size to the storage block shown in Figure 3-12. If we proceed to fill the memory bank shown in Figure 3-13 with the elements located in the storage block shown in Figure 3-11 the elements e_5, e_4, e_3, e_2 and e_1 will be arranged in ascending order from right to left.

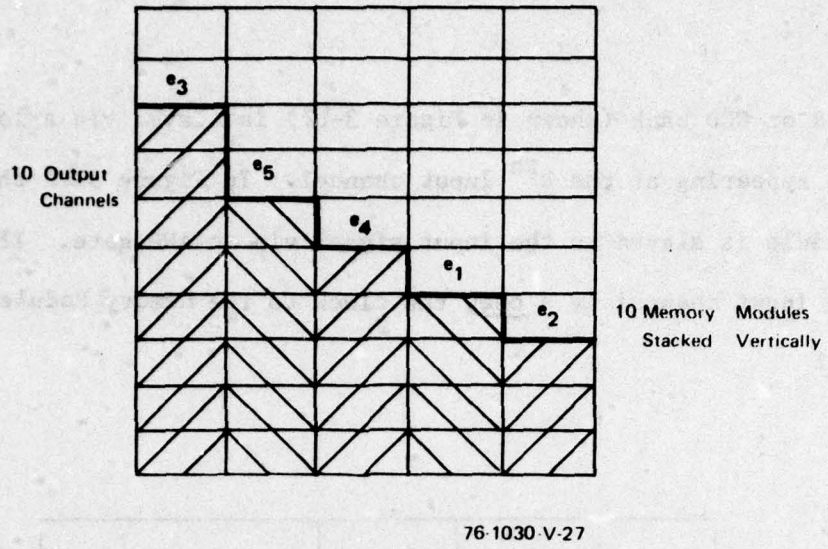


Figure 3-14. Memory Bank Consisting of 10 Memory Modules (Median Extracting Module)

The ordering of the elements e_5, e_4, e_3, e_2 and e_1 occurs because of the manner in which the clock of each memory module is slaved to the input signal. For example, the first element e_1, will be shifted into the memory bank's lowest four memory modules into the S_5 position. Next, the e_2 element will appear at the memory bank's input. Only the lowest two memory module's clocks will be clocked hence the loaded elements e_2 and e_1 will be ordered in the memory bank such that e_1 will be located on the left of e_2. Continuing with the elements e_3, e_4 and e_5 it is easy to show that all the elements will be ordered in the memory block. Clearly if nondestructive taps are placed on the middle (i.e., location S_3) of the memory bank the median value of the elements e_1, e_2, e_3, e_4 and e_5 can readily be obtained. Moreover, if the elements exceeding the 20% threshold level are sought a nondestructive read out structure attached to the S_1 position of the memory bank will yield this level. The size of the memory bank (in quantized levels and data elements) can be readily extended to accommodate larger data sets. Similarly levels other than the median can be readily extracted.

The method for finding a median of a data ensemble has been outlined. The composition of the outlined median extracting module (MEM) is such that a complete data ensemble has to be loaded into the memory bank (see Figure 3-14) before the median is obtained. However with some modifications, the median extracting module can yield the median of N elements selected by a moving window N elements wide. In this operation the difference between successive sets of elements located within the moving windows, one element time apart, will be a single element. It is desirable to compute the median of each successive element set by utilizing the signal processing already performed on the N-1 elements in common between the current element set and its predecessor. This can be performed if we simply remove from the median extracting module the "old" element and replace it with the "new" element. Mechanization of this requires several modifications to the original median extracting module: knowledge of the "old" and "new" elements in the element data set, and a memory module with a slaved clock which can shift the stored bits left or right. The "old" element can be obtained if an additional delay storage block (e.g., see Figure 3-12) is used to delay the signals applied to the input of the median extracting module. The size and delay time provided by the delay storage block should be equal to the size and delay time of the median extracting module. Hence data which are concurrently fed into the median extracting module and the delay storage block (N elements wide) will emerge out of the storage block in phase with the moving window N elements wide.(See Fig. 3-12 Particularly the "old" element will be emerging from the delay storage block when the "new" element will be entering the median extracting module and the delay storage block. Knowing the "old" and "new" elements, the clocks for each memory module (e.g., see Figure 3-13), composing the median extracting

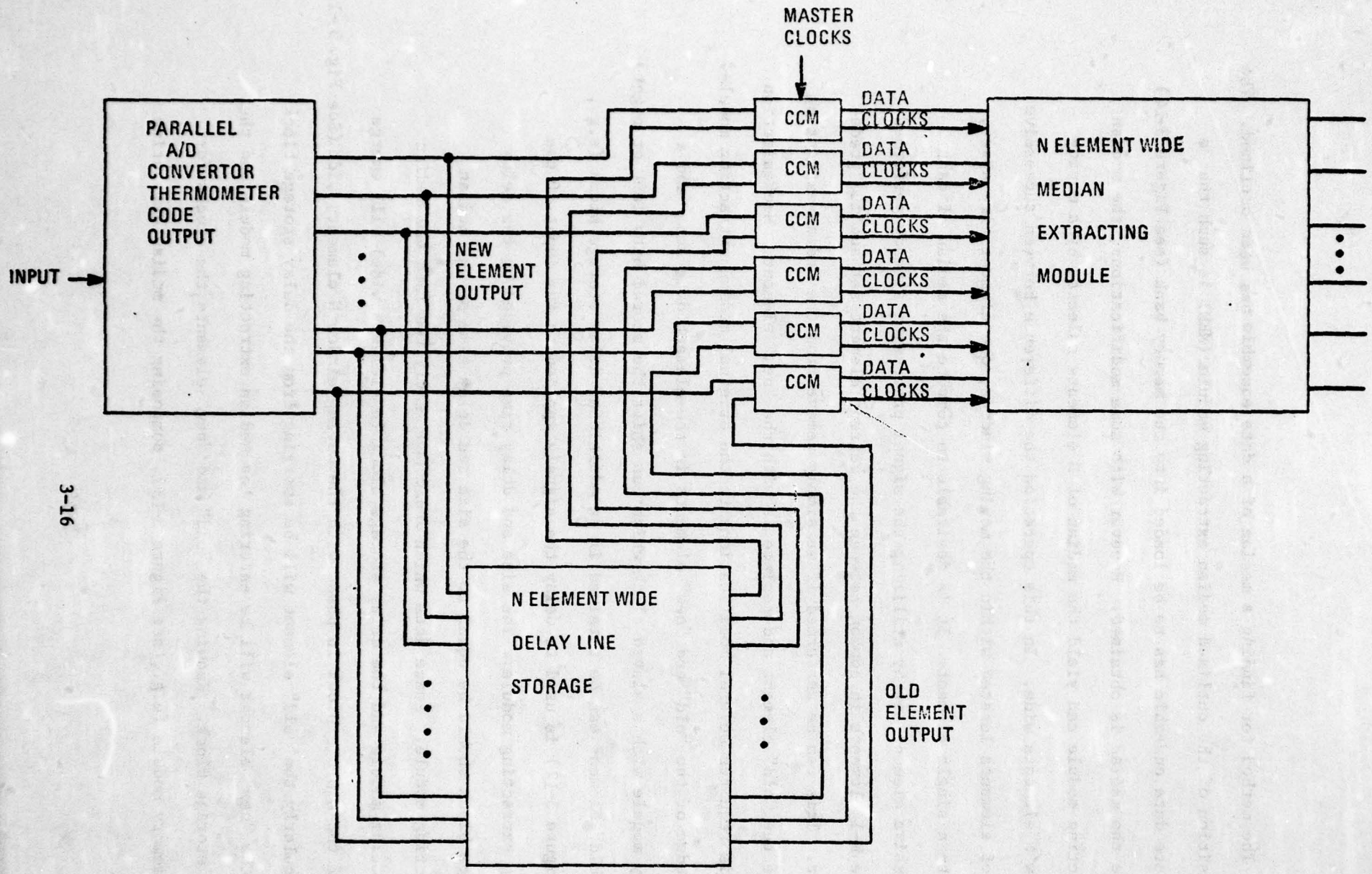


Fig. 3-12a. Block Diagram for Moving Window Median Extracting Module

module, can be slaved to remove the old element and insert the new element into the MEM. The function by which the clock of the MM will be governed is referenced in Figure 3-12a by the Clock Control Module (CCM) and can be mathematically expressed as follows:

$$(old)_{ij} \cap (New)_{i,J+N} = 1 \qquad \text{Clock disabled}$$

$$(Old)_{ij} \cup (New)_{i,J+N} = 0 \qquad \text{Clock disabled} \qquad (1)$$

$$(Old)_{ij} \cap \overline{(New)}_{i,J+N} = 1 \qquad \text{Clock shift left}$$

$$\overline{(Old)}_{ij} \cap (New)_{i,j+N} = 1 \qquad \text{Clock shift right}$$

The subscripts (ij) refer respectively to the i input row of the MEM and the second subscript refers to the particular element from the ensemble. Note that the "New" and "Old" elements are separated by N data elements. Equation 1, governing the clock, can be described as follows: if the $(Old)_{i,j}$ and $(New)_{i,j+N}$ bits are equal no shifting is necessary, when the $(Old)_{i,j}$ bit is greater than the $(New)_{i,j+N}$ bit value the MM should shift left and for the condition when the $(Old)_{i,j}$ bit is less than the $(New)_{i,j+N}$ bit the clock should shift the MM right. Embodiment of the clock control slowed to the $(New)_{i,j+N}$ and $(Old)_{i,j}$ input can be realized as shown in Figure 3-15. Two different clocks are necessary for shifting the MM left or right depending on the input signals.

At the end of each line, the median array accumulates the last columns based on the old row and the first columns based on the new row. Under this mixed condition, median calculations are suspended.

Both Westinghouse and Hughes have approached the histogram problem, based on semiannual reports, from an analog-digital hybrid direction. However, the Westinghouse implementation seems more able to accommodate a moving window as found in the Median Filter.

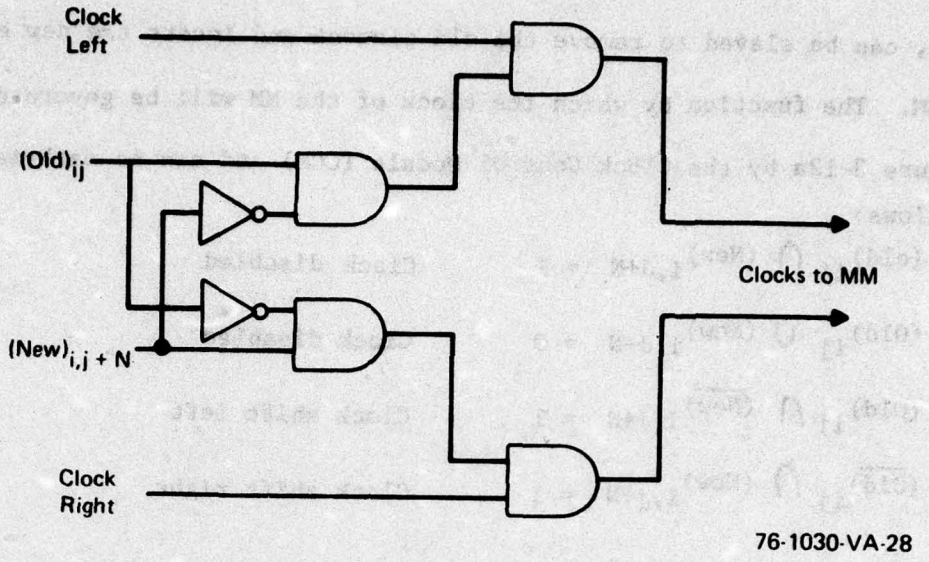


Figure 3-15. Control for Memory Module Clocks (CCM)

4.0 FUTURE DIRECTIONS

In line with our emphasis on focal plane processing, we want to continue the work on analog implementation of the histogram.

Also, we want to examine the cost to fabricate, yields, and form of a possible configuration for the Gradient Operator of the Threshold Algorithm.

We shall be examining the effectiveness of algorithms such as Median Filter within the context of an entire cueing system.

A final item is to analyze the proposed line growing techniques of Maryland for possible implementation by CCD's.